

PATENT OFFICE
JAPANESE GOVERNMENT

JC925 U.S. PTO
09/670424
09/26/00

This is to certify that the annexed is a true copy of the following application as filed with this Office.

Date of Application: July 25, 2000

Application Number: Patent Application
No. 2000-224534

Applicant(s): CASIO COMPUTER CO., LTD.

August 25, 2000

Commissioner,
Patent Office Kozo Oikawa

Certificate No. 2000-3066887

日本国特許庁

PATENT OFFICE
JAPANESE GOVERNMENT



別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出願年月日
Date of Application:

2000年 7月25日

出願番号
Application Number:

特願2000-224534

出願人
Applicant(s):

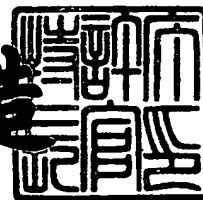
カシオ計算機株式会社

CERTIFIED COPY OF
PRIORITY DOCUMENT

2000年 8月25日

特許庁長官
Commissioner,
Patent Office

及川耕造



出証番号 出証特2000-3066887

【書類名】 特許願

【整理番号】 00-0287-00

【提出日】 平成12年 7月25日

【あて先】 特許庁長官殿

【国際特許分類】 G09C 1/00

【発明者】

【住所又は居所】 東京都羽村市栄町3丁目2番1号 カシオ計算機株式会社
社羽村技術センター内

【氏名】 竹田 恒治

【特許出願人】

【識別番号】 000001443

【氏名又は名称】 カシオ計算機株式会社

【代理人】

【識別番号】 100093632

【弁理士】

【氏名又は名称】 阪本 紀康

【電話番号】 03-3238-0058

【選任した代理人】

【識別番号】 100074099

【弁理士】

【氏名又は名称】 大菅 義之

【電話番号】 03-3238-0031

【手数料の表示】

【予納台帳番号】 012900

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【包括委任状番号】 9004585

【包括委任状番号】 0003549

【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 暗号化方式及び復号化方式

【特許請求の範囲】

【請求項 1】 n ($n \geq 1$) 次元の空間の閉領域内に定義されたベクトルの各成分とパラメータ集合 P によって決まる角度 Ω_n を用いて、該ベクトルを回転させる n 次の回転行列 $R_n (\Omega_n)$ を、 $n-1$ 次の回転行列 $R_{n-1} (\Omega_{n-1})$ を $n-1$ 次小行列として含むように生成する回転行列生成手段と、

少なくとも前記回転行列 $R_n (\Omega_n)$ を含む非線形関数を用いて逐次生成される各ベクトル r_j ($j \geq 0$) が、前記 n 次元空間内で一致しないように該ベクトル r_j を生成するベクトル生成手段と、

原文データと前記ベクトル生成手段によって生成されたベクトル r_j の成分との間で 2 項演算を行って暗号化データを生成する 2 項演算手段と、

を含むことを特徴とする暗号化方式。

【請求項 2】 前記ベクトル生成手段の前記非線形関数は、回転ベクトルを並進させる固定ベクトルを含む関数であり、前記ベクトル生成手段は、生成される各ベクトルが一致することがないように、該各ベクトルを逐次的に生成することを特徴とする請求項 1 記載の暗号化方式。

【請求項 3】 前記ベクトル生成手段が使用する n 次の前記回転行列 $R_n (\Omega_n)$ は、 $n-1$ 次の回転行列 $R_{n-1} (\Omega_{n-1})$ に対応する $n-1$ 次小行列の挿入場所を変更して生成される n 個の n 次回転行列の積によって生成されることを特徴とする請求項 1 記載の暗号化方式。

【請求項 4】 前記 2 項演算 ($\circ p$) はスクランブル操作 S を行った後に排他的論理和操作 (XOR) を行うことを意味する

$$\circ p = XOR \cdot S,$$

であることを特徴とする請求項 1 記載の暗号化方式。

【請求項 5】 前記ベクトル生成手段が使用する非線形関数によって生成される j 番目のベクトル r_j と $j-1$ 番目で生成された暗号化データ C_{j-1} のチェックサム Σ_{j-1} との前記 2 項演算を行ってできるベクトルに更に原文データ M_j に対して前記 2 項演算を行って暗号化データ C_j を形成することを特徴とする請

求項 1 記載の暗号化方式。

【請求項 6】 n ($n \geq 1$) 次元空間の閉領域内に定義されたベクトルの各成分とパラメータ集合 P によって決まる角度 Ω_n を用いて前記ベクトルを回転させる n 次の回転行列 R_n (Ω_n) を少なくとも含む非線形関数を用いて逐次生成される各ベクトル r_j が、前記 n 次元空間内で一致しないように該ベクトル r_j を生成するベクトル生成手段と、

暗号側から、原文データと前記ベクトル生成手段と同様な方法によって生成されたベクトル r_j の成分との間で 2 項演算を行って生成される暗号化データを受信し、前記ベクトル生成手段によって生成された前記ベクトル r_j と前記暗号化データを用いて、前記 2 項演算の逆操作に対応する逆 2 項演算を行って前記原文データを復号する逆 2 項演算手段と、

を具備することを特徴とする復号化方式。

【請求項 7】 前記回転行列 R_n (Ω_n) は、請求項 1 記載の回転行列生成手段によって生成されることを特徴とする請求項 6 記載の復号化方式。

【請求項 8】 前記ベクトル生成手段が使用する前記非線形関数は回転ベクトルを並進させる固定ベクトルを含む関数であり、前記ベクトル生成手段は、生成される各ベクトルが一致することがないように、該各ベクトルを逐次的に生成することを特徴とする請求項 6 記載の復号化方式。

【請求項 9】 前記ベクトル生成手段が使用する n 次の前記回転行列 R_n (Ω_n) は、 $n-1$ 次の回転行列 R_{n-1} (Ω_{n-1}) に対応する $n-1$ 次小行列の挿入場所を変更して生成される n 個の n 次回転行列の積によって生成されることを特徴とする請求項 6 記載の復号化方式。

【請求項 10】 前記 2 項演算 (\circ_p) はスクランブル操作 S を行った後に排他的論理和操作 (XOR) を行うことを意味する

$$\circ_p = XOR \cdot S,$$

であって、

前記逆 2 項演算 (\circ_p^{-1}) は、排他的論理和操作 (XOR) を行った後に前記スクランブル操作 S の逆操作 S^{-1} を行うことを意味する

$$\circ_p^{-1} = S^{-1} XOR$$

であることを特徴とする請求項 6 記載の復号化方式。

【請求項 11】 $j-1$ 番目の受信暗号化データ C_{j-1} のチェックサム Σ_{j-1} を生成し、次に、その生成結果と前記ベクトル生成手段が使用する前記非線形関数によって生成されるベクトル r_j との前記 2 項演算を行い、更に、その 2 項演算によって生成されるベクトルと j 番目の受信暗号化データ C_j に対して前記逆 2 項演算を実行して原文データ M_j を復号することを特徴とする請求項 6 記載の復号化方式。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明は、例えば暗号通信を行うシステムに用いられる暗号復号化方式に係り、特に多次元ベクトルを用いて原文データを暗号・復号化する方式に関する。

【0002】

【従来の技術】

計算機やネットワークをはじめとする情報システムでは、不特定多数のユーザが使用する環境下において、悪意のユーザによる情報の盗聴や改竄などが大きな問題となっており、その対策として暗号復号化技術を利用することが重要視されている。

【0003】

暗号復号化技術として一般的に知られている手法として、例えば、下記の文献に示されているものが知られている。

文献名：「Communications of the ACM Vol. 21, No. 2 (1978)、p120.、A Method for Obtaining Digital Signatures and Public-Key Crypto Systems: R. L. Rivest, A. Shamir, and L. Adleman, MIT Laboratory for Computer Science and Department of Mathematics.」

この文献で発表された暗号化手法は、一般に、かなり強力なものとして受入れ

られ、RSA (R i v e s t - S h a m i r - A d l e m a n) 法と呼ばれている。また、このRSA法から派生した手法が電子商取引におけるデータの暗号化及び「著名」の認証手段として開発され、実用化されつつある。

【0004】

RSA法は、素因数分解の困難さを利用した公開鍵（非対称）暗号方式であり、データを累乗した結果を大きな整数で割った余り（剰余）を暗号文にする。このRSA法の特徴は、整数データに対する剰余系の範囲で利用されること、そして、2つの素数（ p および q ）の積から元の2つの素数を見つけ出すのが困難で、たとえ、2つの素数の積が知られたとしても、 p および q 、更には解読演算を推定することが困難であるということを利用している点である。

【0005】

【発明が解決しようとする課題】

前述したRSA法自体は実用的であり、信頼性も、暗号鍵としてのデータのビット長が十分である場合には、かなり高いものになることが保証されている。この信頼性を確保するために、現在は256ビット長の暗号鍵データを使うのが主流となっているが、ときとして、これでは短かすぎ、最近では、さらに512ビットや1024ビットのデータ長の暗号鍵の必要性が論じられている。

【0006】

しかしながら、整数データ長は計算機の演算精度と演算速度で事実上制限されるため、やたらにビット長を長くすることは、実用上、必ずしも得策とは言えない。また、逆に、ビット長を短くした場合には、信頼性がかなり低下する。

【0007】

また、RSA法から派生させられた暗号復号化手法は、整数データによる剰余系で成立する理論をベースにしているため、システムの性能によって、その信頼性が制約されるといった問題点を抱えていた。

【0008】

本発明は、このような問題点に鑑みなされたもので、デジタル表現あるいはアナログ（実数）表現される任意表現データに対して暗号復号化を行うことができ、これにより、広範囲なアプリケーションに対して使用可能な汎用性に優れ、

かつ、信頼性にも優れた暗号復号化方式を提供することを目的とする。

【0009】

【課題を解決するための手段】

図1は、本発明の原理を説明する図である。

同図において、送信側と受信側のそれぞれの装置10、12の内部にあるセキュリティ・デバイス、すなわち暗号化／復号化エンジンには共通鍵が記憶されている。一方の装置10から他方の装置12に暗号通信を行う場合、装置10上の主プログラムは制御をセキュリティ・デバイス上の副プログラムまたは専用ハードウェアに渡す。

【0010】

送信側の暗号化処理を行うセキュリティ・デバイスでは、共通鍵に対応するパラメータに従って変更可能な非線形関数を利用する。すなわち、 n 次元の空間の閉領域内に定義された n 次元ベクトルを回転並進させる非線形関数を用いてカオス的かつ逐次的にベクトルを生成し、原文データと前記生成されたベクトルとの論理演算をビット単位で行って暗号化データを生成する。

【0011】

又、受信側の復号化処理を行うセキュリティ・デバイスでは、前記ベクトルを送信側と同様にして生成し、これに逆操作を実行し、受信した暗号データから元の原文データを簡易に復号する。

【0012】

本発明では、前記多次元ベクトル生成する非線形関数を決定するパラメータが第三者に秘密となる。これにより、本発明では、前記 n 次元ベクトルの生成は、少なくとも共通鍵（秘密鍵）を定義して決定され、それによって、生成される各 n 次元ベクトルが互いに一致することがないようにカオスを発生することが可能な非線形関数を用いて該 n 次元ベクトルを逐次的に生成する。

【0013】

すなわち、本発明は、 n ($n \geq 1$) 次元の空間の閉領域内に定義されたベクトルの各成分とパラメータ集合 P によって決まる角度 Ω_n を用いて、前記ベクトルを回転させる n 次の回転行列 R_n (Ω_n) を少なくとも含む非線形関数を用いて

逐次生成される各ベクトルが前記 n 次元空間内で一致しないようにベクトル r_j を生成するベクトル生成手段と、暗号化処理において、原文データと前記ベクトル生成手段によって生成されたベクトルの成分との間で 2 項演算を行って暗号化データを生成する 2 項演算手段を具備し、復号化処理において、前記ベクトル生成手段によって生成されたベクトル r_j と前記暗号化データを用いて前記 2 項演算の逆操作に対応する逆 2 項演算を行って前記原文データを生成する逆 2 項演算手段を具備することを特徴とする。

【 0 0 1 4 】

特に、 n ($n \geq 1$) 次元の空間の閉領域内に定義されたベクトルの各成分とパラメータ集合 P によって決まる角度 Ω_n を用いて、該ベクトルを回転させる n 次の回転行列 $R_n (\Omega_n)$ を、 $n-1$ 次の回転行列 $R_{n-1} (\Omega_{n-1})$ を $n-1$ 次小行列として含むように生成する回転行列生成手段と、少なくとも前記回転行列 $R_n (\Omega_n)$ を含む非線形関数を用いて逐次生成される各ベクトル r_j ($j \geq 0$) が、前記 n 次元空間内で一致しないように該ベクトル r_j を生成するベクトル生成手段と、原文データと前記ベクトル生成手段によって生成されたベクトル r_j の成分との間で 2 項演算を行って暗号化データを生成する 2 項演算手段と、を含むことを特徴とする。

【 0 0 1 5 】

【発明の実施の形態】

以下、図面を参照して本発明の一実施形態を説明する。

本発明は、データの送信者、受信者が共通のセキュリティ・デバイス（暗号化装置）を持ってデータ通信する場合での暗号復号化処理に関するものである。

【 0 0 1 6 】

本暗号方式においては、データ送信者（暗号側）において、平文メッセージを所定の共通鍵に基づいて生成された鍵系列を用いて、ビット単位の所定の論理演算（一般的には、排他的論理和演算）を施して暗号文を生成し、データ受信者（復号側）において、暗号文を所定の共通鍵（暗号側で使用した鍵と同一のもの）に基づいて生成された暗号側と同一の鍵系列を用いて、ビット単位の所定の論理演算（暗号側で使用した演算と同一のもの）を施して元の平文メッセージを得る

【 0 0 1 7 】

そして、このような暗号方式において、上記の鍵系列を生成する乱数生成装置として、本発明では多次元ベクトル生成装置を用いる。この場合、多次元ベクトル生成装置のベクトル生成関数を定める各種パラメータや初期状態は、共通鍵として与えられるようにする。

【 0 0 1 8 】

図 1 に、本発明を適用した暗号システムの構成例を示す。暗号側の装置 1 0 に、多次元ベクトル生成機能部 1 0 1 と論理演算処理機能部 1 0 2 が備えられている。

【 0 0 1 9 】

同様に、復号側の装置 1 2 にも、多次元ベクトル生成機能部 1 2 1 と論理演算処理機能 1 2 2 が備えられている。

図 1 において、暗号側の装置（暗号化装置） 1 0 と復号側の装置（復号化装置） 1 2 との間では、機密状態の下で、例えば、ＩＣカード等により共通鍵が配布され、共通鍵を共有しているものとする。そして、暗号側の装置 1 0 では、所定の共通鍵により決定される関数に基づいて多次元ベクトルを生成し、該ベクトルの成分データを乱数系列として、平文メッセージとの排他的論理和を取って、平文メッセージを暗号文に変換し、それをネットワークを介して復号側の装置 1 2 に送信する。

【 0 0 2 0 】

暗号文を取得した復号側の装置 1 2 では、この暗号文について、暗号側の装置 1 0 が備える多次元ベクトル生成機能部 1 0 1 と同等の機能を有する多次元ベクトル生成機能部 1 2 2 によりベクトルを生成し、このベクトルと上記のようにして生成された乱数系列との排他的論理和を取ることで、元の平文メッセージを復元する。

【 0 0 2 1 】

また、暗号装置 1 0、復号装置 1 2 で行う処理は実質的に同じであるので、実際には、計算機などの処理装置は、暗号装置 1 0 と復号装置 1 2 の両方の機能を

持つことができる。

【 0 0 2 2 】

図 2 は、暗号化・復号化装置 1 0、1 2 が有する暗号化・復号化のためのプログラム構成を示す。

主プログラム 1 は、データの入出力の管理を行い、データが暗号化あるいは復号化が必要なデータであるかの判定を行い、また、暗号化・復号化の全体処理の管理を行うプログラムで構成される。パラメータリスト生成ライブラリ 2 は、I C カード等により配布された共通鍵を記憶しており。暗号化／復号化エンジン 3 は、パラメータリスト生成ライブラリ 2 から共通鍵をパラメータとして受け取って、多次元ベクトル回転関数発生ライブラリ 4 が決定した行列に基づき回転ベクトルを発生し、該ベクトルの成分を用いて平文の暗号化、あるいは暗号文の復号化を行う。

【 0 0 2 3 】

以下では、多次元ベクトルの生成について説明して行く。

多次元 (n 次元) 空間内に定義されたベクトル r_{j-1} に対する回転を考え、一般化された回転角を Ω_n で表わし、この回転に対応する操作を $n \times n$ の行列として $R_n (\Omega_n)$ と表わす。すなわち、 $R_n (\Omega_n)$ は r_{j-1} に作用してこのベクトルを回転させる。ここで、次式 (1) により新しいベクトル r_j を定義する。

【 0 0 2 4 】

$$r_j = a R_n (\Omega_n) r_{j-1} + c \quad (1)$$

ここに、 a は定数であり、 $|a| \leq 1$ を満足するものとする。また、 c は、 n 次元の定数ベクトルである。上式はベクトル r_{j-1} の回転と並進により新しいベクトル r_j が生成されることを意味している。

【 0 0 2 5 】

本発明では、回転角 Ω_n に r 依存性を持たせる事により、発生される r ベクトルの系列が、カオス的、すなわち、閉じられた空間の中で発生系列が元の系列にならないように、非線形な系列を発生するようにする。すなわち、 Ω_n は形式的に次式 (2) のようにパラメータ P とベクトル r の関数として表わされる。

【 0 0 2 6 】

$$\Omega_n = \Omega_n (P, r_{j-1}) \quad (2)$$

ここで、 P は Ω_n を与える関数の中で使われている、任意個数のパラメータの集合を意味する。

【0027】

$$P = \{p_i \mid i = 1, 2, 3, \dots\} \quad (3)$$

これは、例えば、2次元のベクトルを考えた場合、2次元の回転角 Ω_2 が2次元ベクトル $r = (x, y)$ の成分 x と y で、

$$\Omega_2 = p_1 x + p_2 y + p_3$$

と表現されることを意味する（上式で、パラメータ p_1 、 p_2 、 p_3 は任意に与えられる）。

【0028】

このような2次元ベクトルを図1のシステムで使用される装置10、12で処理する場合の具体的な動作を、図3のフローチャートを用いて説明する。

2次元ベクトル r は、直行座標系の成分 x 、 y とを用いて、 $r = (x, y)$ と表される。このベクトルに対する角度 $\Omega_2 = \theta$ の回転操作は、

【0029】

【数1】

$$R_2(\theta) = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}$$

【0030】

と2次元行列で表される。

回転角を求める関数 $\theta = p_1 x + p_2 y + p_3$ 及び求められた回転角でベクトルを回転操作する関数が、多次元ベクトル関数発生ライブラリ4（図2参照）に記憶されているとし、 r_0 の初期値 x_0 、 y_0 及び P の値、 $p_1 = 1$ 、 $p_2 = 1$ 、 $p_3 = 1$ が共通鍵として予めパラメータリスト生成ライブラリ2（図2参照）に設定されている場合を例にして説明する。

【0031】

2次元ベクトルを生成するには、まず、初期値としての r_0 （成分データ x_0

、 y_0 を含む) 及び回転角 θ を定義する関数のパラメータ p_1 、 p_2 、 p_3 をパラメータリスト生成ライブラリ 2 (図 2 参照) から読みだして装置 (10、12) のメモリのワークエリアに記憶する (ステップ 21)。そして、 r_0 の値 x_0 、 y_0 を元にして、角度 $\theta = p_1 * x_{j-1} + p_2 * y_{j-1} + p_3$ ($\theta = p_1 * x_0 + p_2 * y_0 + p_3$) を計算し、その計算結果を値 θ として記憶する (ステップ 22)。

【0032】

次に、回転行列 R の要素の値を決めるために、 $\cos \theta$ と $\sin \theta$ を求め、それらの値を、それぞれ、 \cos_t 、 \sin_t として記憶する (ステップ 23)。

そして、次に、新しいベクトル r_j を $r_j = a R_2(\Omega_2) r_{j-1} + c$ で計算して生成する (ステップ 24)。つまり、

$$x_j = a * (\cos_t * x_{j-1} - \sin_t * y_{j-1}) + c_x;$$

$$y_j = a * (\sin_t * x_{j-1} + \cos_t * y_{j-1}) + c_y;$$

を計算し、新しいベクトル r_j (成分 x_j 、 y_j) を生成する。

【0033】

そして、このベクトル r_j の成分を元に次の回転角 θ を求め (ステップ 22)、前述のステップ 23、24 を繰り返すことで逐次的にベクトルを発生させる。

本発明ではこのように、回転を使うことにより三角関数が導入され、三角関数の積が使われる為、通常のカオス関数と比べて非線形性が増し、より解読が困難になるという特徴を備えている。

【0034】

次に、多次元ベクトルの生成を利用したデータの暗号化について述べる。

図 4 に示すように、暗号化処理にあっては、まず、 n 次元回転行列 $R_n(\Omega_n)$ を生成する (ステップ 41)。尚、この行列の生成方法の詳細については後述する。

【0035】

そして、この n 次元回転行列 $R_n(\Omega_n)$ を含む非線形関数を用いてベクトルを生成する (ステップ 42)。ここで生成される各ベクトル r_j は、前記 n 次元空間内で一致しないように逐次生成される。次に、平文データと前記ベクトル生

成手段によって生成されたベクトルの成分との間で2項演算を行って暗号化データを生成する（ステップ43）。そして、上記暗号化データを相手方受信装置に送信する（ステップ44）。

【0036】

ここで、上記ステップ43における2項演算について説明する。

逐次生成されるベクトル r が N ビットで表現されるものとする。例えば、成分 x 、 y で表される2次元ベクトルの場合、 x 、 y のデータ値がそれぞれ16ビットで示されるものとし、 x 、 y のデータを並べて N ビット（例えば、32ビット）とする。

【0037】

この手続きによって得られるベクトル列 r_j ($j = 1, 2, 3, \dots$) と暗号化されるべき平文データ M を N ビット毎に分けたデータ列 M_j ($j = 1, 2, 3, \dots$) を2項演算子 \circ_p として排他的論理和 (XOR) で処理した結果 C_j ($j = 1, 2, 3, \dots$) が暗号化データとして得られる。すなわち、

$$C_j = r_j \circ_p M_j \quad (5)$$

上記2項演算子 \circ_p としては、各ビット毎の排他的論理和 (XOR) を使用するのが一般的であるが、排他的論理和は可逆性の性質を有することから、これをそのまま暗号化のための演算子として使うことは危険である。そこで、排他的論理和のこの欠点を補うために、2項演算子 \circ_p として、排他的論理和と M_j のビットをごちゃ混ぜにする操作 (scramble) とを併用することが好ましい。

【0038】

その場合は、

$$\circ_p = \text{XOR} \cdot S \quad (6)$$

となる。ここで、 S は M_j のビットをごちゃ混ぜにするスクランブル操作を意味し、 $\text{XOR} \cdot$ はその後の排他的論理和の操作として定義されることを意味する。そして、 $C_j = r_j \circ_p M_j$ で暗号化データを求めるようにする。

次に、復号化処理について図4を参照して説明する。

【0039】

復号化処理にあつては、暗号化と同様に、まず、 n ($n \geq 1$) 次元の空間の閉領域内に定義されたベクトルを回転させる回転行列 R_n (Ω_n) を生成する (ステップ 4 5)。そして、その回転行列 R_n (Ω_n) を含む非線形関数を用いて生成される各ベクトルが前記 n 次元空間内で一致しないようにベクトル r_j を逐次生成する (ステップ 4 6)。

【0 0 4 0】

次に、受信した暗号化データと前記ベクトル生成ステップ 4 6 によって生成されたベクトル r_j の成分との間で前記ステップ 4 3 における 2 項演算の逆操作に対応する逆 2 項演算を行って復号化データを生成する (ステップ 4 7、4 8)。

【0 0 4 1】

この復号化処理においては、受信した暗号化文字列 C_j ($j = 1, 2, 3, \dots$) を順番に取り出して、 C_j に対応するベクトルを生成しながら復号演算をするが、この処理を図 5 のフローチャートを用いて説明する。

復号化処理は $j = 0$ からスタートし (ステップ 5 1)、暗号化データ C_j を取り出し (ステップ 5 2)、 n 次元回転行列 R_n (Ω_n) を生成し (ステップ 5 3)、次にベクトル r_j を生成する (ステップ 5 4)。そして、 $M_j = r_j \circ p^{-1} C_j$ を演算して復号化データ (平文 M_j) を産出する (ステップ 5 5)。そして、暗号データ C が終了していないなら、 $j = j + 1$ として (ステップ 5 6、5 7)、次の暗号データを取り出して、 R_n (Ω_n) を生成し、次のベクトル r_j を生成する処理を繰り返す。このステップ 5 2 ~ 5 6 の処理の繰り返しは、暗号データ C_j が終了するまで行う。

【0 0 4 2】

ここで、 $\circ p^{-1}$ は $C_j \text{ XOR } r_j$ の演算後に、暗号化で行ったスクランブル S の逆操作 S^{-1} を行う演算である。

次に、前述した第 1 の暗号化・復号化の実施例を、より実際に近い暗号化処理の手続きに拡張した第 2 の実施例を説明する。

【0 0 4 3】

まず、 $C_0 = r_0 \circ p \quad M_0$ (7)

を求める。次に C_0 に対するチェックサム Σ_0 を計算する。更に、 $j \geq 1$ であるような j に対し、式(5) を次のように書き換える。

【0044】

$$C_j = (r_j \text{ op } \Sigma_{j-1}) \text{ op } M_j \quad (8)$$

チェックサムは、例えば算出された C の値に含まれる 1 の数を r_j のビット数と同じビット数の 2 進で表現したものであり、 Σ_0 は暗号化データ C_0 の値から、 Σ_1 は C_1 から、 Σ_2 は C_2 から得るもので、次のような計算順序に従うことになる。

【0045】

【表 1】

平文	M_0	M_1	M_2	M_3	...
(鍵列)	r_0	r_1	r_2	r_3	...
暗号	C_0	C_1	C_2	C_3	...
	Σ_0	Σ_1	Σ_2	Σ_3	...

\nearrow \nearrow \nearrow

【0046】

すなわち、送信側は M_0 に対する暗号化データ C_0 については、 $C_0 = r_0 \text{ op } M_0$ で C_0 を算出し、これに伴い C_0 のチェックサム Σ_0 を求める。 M_1 に対する暗号化データ C_1 については、 $C_1 = (r_1 \text{ op } \Sigma_0) \text{ op } M_1$ で C_1 を算出する。そして C_1 のチェックサム Σ_1 を求める。以降のデータ M_j は、その前のデータで求めた Σ_{j-1} を加味し、 $C_j = (r_j \text{ op } \Sigma_{j-1}) \text{ op } M_j$ で暗号化する。ここで、 r_j と Σ_{j-1} は同じデータ幅（ビット数）で算出する。

【0047】

復号するべき受信側では、 C_0 、 C_1 、 C_2 ・・・を受信して、まず C_0 を用いて $M_0 = r_0 \text{ op }^{-1} C_0$ を計算するが、その、受信した C_0 からチェックサム Σ_0 を求める必要がある。これを使って、 C_1 については

$$M_1 = (r_1 \text{ op } \Sigma_0) \text{ op }^{-1} C_1 \quad (9)$$

で M_1 を計算する。

【0048】

そして以降のデータ M_j については、受信した C_{j-1} について求めたチェックサム Σ_{j-1} を利用し、

$$M_j = (r_j \text{ op } \Sigma_{j-1}) \text{ op}^{-1} C_j \quad (10)$$

を用いて復号化していくことになる。

【0049】

このような手続きによって暗号化されたデータは、「鍵の違い」が伝播していくため、鍵を仮定して復号化しようとする攻撃に耐性を持つといえる。

次に、多次元空間の回転について（回転行列 $R_n(\Omega_n)$ の導出）について説明する。

【0050】

多次元空間の回転を扱う方法は複雑なので、まず、簡単なため、2次元の場合から考える。2次元ベクトル r は直交座標系の成分 x と y とを用いて

$$r = (x, y) \quad (11)$$

と表される。このベクトルに対する角度 $\Omega_2 = \theta$ の回転操作は、

【0051】

【数2】

$$R_2(\theta) = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \quad (12)$$

【0052】

と2次元行列で表される。ここに、左辺の添字2は操作が2次元空間で定義されていることを意味する。この操作は次式の性質を満たしている。

$$|R_2(\theta)| = 1 \quad (13)$$

$$R_2(-\theta) = R_2(\theta)^{-1} \quad (14)$$

式(13)は回転操作によって回転したベクトルの大きさが不変に保たれることを保証し、式(14)は回転したベクトルを元に戻す回転操作が存在することを意味している。

【0053】

3次元回転への拡張のため、式(12)の右辺の表記を簡略化し、形式的に

【 0 0 5 4 】

【数 3】

$$R_2(\theta) = \begin{bmatrix} \alpha_{11} & \alpha_{12} \\ \alpha_{21} & \alpha_{22} \end{bmatrix} \quad (15)$$

【 0 0 5 5 】

と表すことにする。ここに、 $\alpha_{11} = \alpha_{22} = \cos \theta$ 及び $\alpha_{21} = -\alpha_{12} = \sin \theta$ である。3次元回転を扱う場合、三つの直交軸の各々を回転軸とした回転から出発するのが妥当であろう。それは、以下の三つの行列

【 0 0 5 6 】

【数 4】

$$R_{3,1}(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \alpha_{11} & \alpha_{12} \\ 0 & \alpha_{21} & \alpha_{22} \end{bmatrix} \quad (16-1)$$

【 0 0 5 7 】

【数 5】

$$R_{3,2}(\theta) = \begin{bmatrix} \alpha_{11} & 0 & \alpha_{12} \\ 0 & 1 & 0 \\ \alpha_{21} & 0 & \alpha_{22} \end{bmatrix} \quad (16-2)$$

【 0 0 5 8 】

【数 6】

$$R_{3,3}(\theta) = \begin{bmatrix} \alpha_{11} & \alpha_{12} & 0 \\ \alpha_{21} & \alpha_{22} & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (16-3)$$

【 0 0 5 9 】

の内のどれかで表される。これらが、式(15)で与えられる2次元空間の回転操作に、対角成分として1を追加することによって得られることに注意する。また、これらの操作による3次元ベクトルの回転が図8のようになることは明らかである。

【0060】

ここで、上の行列(16-1), (16-2), (16-3)は、2次元空間での回転操作を表す2次元行列を含む3次元の行列となっている。一般化された3次元回転は、これらの行列から重複を許して三つ取り出し、それらを順次掛け合わせることで得られる。3次元空間での一般化された回転角を、 $\Omega_3 = (\theta_1, \theta_2, \theta_3)$ と表せば、3次元ベクトルに対する回転操作 $R_3(\Omega_3)$ は次式で表される。

【0061】

$$R_3(\Omega_3) = R_{3,i}(\theta_i) R_{3,j}(\theta_j) R_{3,k}(\theta_k) \quad (17)$$

ここに、 i, j, k は1, 2, 3のどれかであり、一般には同一軸に対する操作が連続しないことを条件に重複が許される。例えば i, j, k が1, 2, 1でもよい。

【0062】

次に、「逆回転角」を $-\Omega_3 = (-\theta_1, -\theta_2, -\theta_3)$ と表わせば、式(17)の回転操作の逆回転操作は、その意味を考慮して

$$R_3(-\Omega_3) = R_{3,k}(-\theta_k) R_{3,j}(-\theta_j) R_{3,i}(-\theta_i) \quad (18)$$

と表すこともできる。式(17)で定義された回転操作は一般に

【0063】

【数7】

$$R_3(\Omega_3) = \begin{bmatrix} \beta_{11} & \beta_{12} & \beta_{13} \\ \beta_{21} & \beta_{22} & \beta_{23} \\ \beta_{31} & \beta_{32} & \beta_{33} \end{bmatrix} \quad (19)$$

【0064】

の形をしている。ここで、各行列要素は式(12), (15), (16-1)～(16-3)と(1

7)から一意的に決められる。また、 $R_3(\Omega_3)$ に対し、以下の二つの性質が満たされることもわかる。

【0065】

$$|R_3(\Omega_3)| = 1 \quad (20)$$

$$R_3(-\Omega_3) = R_3(\Omega_3)^{-1} \quad (21)$$

実際の3次元の場合のベクトルの発生について説明する。

【0066】

3次元の場合は、回転行列の掛け算の順序を記憶することによりベクトル r_j の発生が可能となる。3次元の回転の場合、回転角は、簡単化のために、

$x_{j-1} = x, y_{j-1} = y, z_{j-1} = z$ と表現すれば、

$$\theta_1 = p_{11}x + p_{12}y + p_{13}z + p_{14}$$

$$\theta_2 = p_{21}x + p_{22}y + p_{23}z + p_{24}$$

$$\theta_3 = p_{31}x + p_{32}y + p_{33}z + p_{34}$$

となり、3次元の回転操作 $R_3(\Omega_3)$ は、後述する方法によって、次の3つの回転行列の掛け算、すなわち、

$$R_3(\Omega_3) = R_{3,i}(\theta_i) * R_{3,j}(\theta_j) * R_{3,k}(\theta_k)$$

で表される。

【0067】

ここで、整数 i, j, k は、1、2、3のどれかであり、一般には重複が許される。すなわち、 $R_{3,1}(\theta_1), R_{3,2}(\theta_2), R_{3,3}(\theta_3)$ の掛け算の仕方は $3 \times 2 \times 2 (= 12)$ 通りあり、その掛け算の順序は送り手のパラメータで決まるとする。この暗号化処理において、ベクトル r_j の発生処理のフローは、3次元の場合、図6に示すようになる。

【0068】

すなわち、送り手のパラメータで指定された掛け算の順序に基づき $R_3(\Omega_3)$ を用意する(ステップ61)。そして、ベクトルの初期値 r_0 、回転角 $\theta_1, \theta_2, \theta_3$ を算出するための関数のパラメータ $p_{11} \sim p_{34}$ を記憶する(ステップ61)を記憶する(ステップ62)。次に、 $r_0(r_{j-1})$ の成分(x, y, z)を用いて、

$$\theta_1 = p_{11}x + p_{12}y + p_{13}z + p_{14}$$

$$\theta_2 = p_{21}x + p_{22}y + p_{23}z + p_{24}$$

$$\theta_3 = p_{31}x + p_{32}y + p_{33}z + p_{34}$$

を計算する（ステップ 6 3）。

【 0 0 6 9 】

そして、 R_3 (Ω_3) を計算し、新しいベクトル r_j を式 (1) から生成する。このとき、掛け算の順序の指定については、上記のように、送り手のパラメータ、例えば、送り手の社員番号等から順序を決めるようにしても良い。また、回転行列 R_3 (Ω_3) については、送り手（及び受け手）側が、送信の都度に指定される順番に基づき回転行列 R_3 (Ω_3) を計算するのではなく、予め 1 2 通りの関数を記憶しておいて、その中のどれを用いるかを指定するようにしても良い。

【 0 0 7 0 】

次に、上記 2 次元回転から 3 次元回転への拡張の手続きを、3 次元回転から 4 次元回転への拡張へ適用する方法を説明する。

この場合、式 (19) へ対角成分として 1 を追加することによって、四つの 2 次元行列、 $R_{4,i}$ (Ω_3) ($i = 1, 2, 3, 4$) を得る。すなわち、

【 0 0 7 1 】

【数 8】

$$R_{4,i} (\Omega_3) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \beta_{11} & \beta_{12} & \beta_{13} \\ 0 & \beta_{21} & \beta_{22} & \beta_{23} \\ 0 & \beta_{31} & \beta_{32} & \beta_{33} \end{bmatrix} \quad (22-1)$$

【 0 0 7 2 】

【数 9】

$$R_{4,2}(\Omega_3) = \begin{bmatrix} \beta_{11} & 0 & \beta_{12} & \beta_{13} \\ 0 & 1 & 0 & 0 \\ \beta_{21} & 0 & \beta_{22} & \beta_{23} \\ \beta_{31} & 0 & \beta_{32} & \beta_{33} \end{bmatrix} \quad (22-2)$$

【0 0 7 3】

【数 1 0】

$$R_{4,3}(\Omega_3) = \begin{bmatrix} \beta_{11} & \beta_{12} & 0 & \beta_{13} \\ \beta_{21} & \beta_{22} & 0 & \beta_{23} \\ 0 & 0 & 1 & 0 \\ \beta_{31} & \beta_{32} & 0 & \beta_{33} \end{bmatrix} \quad (22-3)$$

【0 0 7 4】

【数 1 1】

$$R_{4,4}(\Omega_3) = \begin{bmatrix} \beta_{11} & \beta_{12} & \beta_{13} & 0 \\ \beta_{21} & \beta_{22} & \beta_{23} & 0 \\ \beta_{31} & \beta_{32} & \beta_{33} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (22-4)$$

【0 0 7 5】

を得る。

更に、4次元空間での回転角 Ω_4 に対する回転操作を次式で定義する。

$$R_4(\Omega_4) = R_{4,i}(\Omega_{3,i}) R_{4,j}(\Omega_{3,j}) R_{4,k}(\Omega_{3,k}) R_{4,l}(\Omega_{3,l}) \quad (23)$$

$\Omega_{3,i}$ ($i = 1, 2, 3, 4$) は、上で定義した3次元回転角 Ω_3 の異なるものを意味する。

【0 0 7 6】

この定義を繰り返して使うことにより、一般に n 次元空間での回転角 Ω_n に対

する回転操作 $R_n (\Omega_n)$ は、掛け算記号 Π を使って

【0077】

【数12】

$$R_n (\Omega_n) = \prod_{i=1}^n R_{n,i} (\Omega_{n-i,i}) \quad (24)$$

【0078】

と表すことができる。こうして得られる回転操作が次式(25)、(26)の性質を満たしていることは、式(24)の右辺の積の順序まで考慮に入れて、容易に確かめられる。

【0079】

$$| R_n (\Omega_n) | = 1 \quad (25)$$

$$R_n (-\Omega_n) = R_n (\Omega_n)^{-1} \quad (26)$$

n 次元回転行列 $R_n (\Omega_n)$ は、図7に示すフローチャートの実行により生成できる。

【0080】

すなわち、まず、 $k=2$ を設定し(ステップ30)、2次元の回転行列 $R_2 (\Omega_2)$ を生成する(ステップ31)。そして、 k の値が n 未満であるか判断し(ステップ32)、 n に達していなければ k の値を“1”インクリメントし(ステップ33)、 k 次の回転行列 $R_k (\Omega_k)$ を、それが $k-1$ 次の回転行列 $R_{k-1} (\Omega_{k-1})$ を $k-1$ 次小行列として含むようにして生成する(ステップ34)。

【0081】

次に、生成された k 個の k 次回転行列 $R_{kj1}(\theta_{ji})$, $R_{kj2}(\theta_{j2})$, \dots , $R_{kjk}(\theta_{jk})$ の積を取り回転行列 $R_k (\Omega_k)$ を得る(ステップ35)。そして、これらのステップ34、35を $k=2$ から $k=n$ まで繰り返すことによって、 n 次元回転行列 $R_n (\Omega_n)$ を生成できる。

【0082】

本発明の特徴は、空間次元を大きくすればするほど暗号化されたデータの復号化が困難になること、ソフトウェア処理によっても十分に高速処理可能であり、

暗号化と復号化のために特別なハードウェアを必要としないこと、更に、個人用、グループ用等の階層化や復号化の特権（権限）を細かく規定し得ること等々である。

【 0 0 8 3 】

このため、本発明の応用範囲も、個人の秘密データ管理、親展メール守秘、同報通信データ守秘等の多岐にわたることになる。加えて、インターネット環境でのサーバ内データの守秘の確実性を高めることも可能なため、システム管理者や、インターネットサービスプロバイダが利用することも可能である。

【 0 0 8 4 】

本発明では更にパラメータ P と定数ベクトル c とに時間依存性を持たせ、 P を

$$P(t) = \{p_i(t) \mid i = 1, 2, 3, \dots\} \quad (27)$$

とし、 c も $c(t)$ とする。更に、ベクトルの初期値 $r_0(t)$ に時間依存性を持たせることも可能である。

【 0 0 8 5 】

実際の暗号化にあたっては、式(1)で、ベクトルの初期値 r_0 を右辺の r_{j-1} ($j = 1$) に代入する。そして、こうして得られた新しいベクトル r_1 を、式(1)の右辺の r_{j-1} に再度代入するという手続きを繰り返していくことにより、次々に新しいベクトルを順次生成していく。式(27)で表される時間依存性は、同じ元データを異なる時刻に暗号化した場合にも、同一の暗号化データが得られないことを保証している。

【 0 0 8 6 】

式(2)においてパラメータの集合と関数を注意深く決定すれば、式(1)によって順次発生させられるベクトル r_j が平衡解に収敛してしまうことを防止できる。

【 0 0 8 7 】

カオスないしはランダム系を使った暗号は、そこで使われた鍵を知られない限り解読が困難であるといわれている。本発明も、このような性質を引き継いでいる。本発明の特徴は、上記従来の暗号化手法の優れた性質を備えることに加え、自由に暗号化の手続きを変更（カスタマイズ）できるところにある。このことは

、以下の理由による。

【0088】

1. 式(1)の回転操作 R_n (Ω_n)の表現は任意に決定できる。
2. 式(2)の右辺の関数 Ω_n (P, r_{j-1})とそこで使われるパラメータ P は、関数値が発散しない、という条件の下で、任意に設定できる。

【0089】

3. 種々の「初期値」を任意に与えることができる。
4. 任意のベクトルの初期値から出発し、式(1)の操作を任意に繰り返すことにより得られるベクトル r_j を、改めて暗号・復号化に使うベクトルの初期値として再設定することができる。

【0090】

5. 浮動小数点演算を実行する場合、演算結果が数値演算プロセッサやコンパイラに依存するため、復号化するためには、暗号化環境と同一の復号化環境を持つ必要がある。

【0091】

尚、本実施例で述べた手続きは、全て整数化が可能である。この場合、多次元空間を格子で区切り、離散的な格子点の座標によって示されるベクトルが回転と並進によって変化することになる。

【0092】

本発明の多次元回転ベクトル方式による暗号化の手続きは任意性に富み、多次元ベクトル回転操作ひとつをとってみても、一義的に決められるものではなく、暗号(方式)を解読しようと試みる人は、回転生成手段の手法を再現し、且つ、一般化された多次元ベクトル回転角を規定する関数系を同定し、更に、そこで使われたパラメータ(鍵)を正確に発見しなければならない。

【0093】

パラメータ P を鍵とし、回転ベクトル r_j の状態から回転角 Ω_n を求める非線形関数の設定、および回転行列の構成の決め方は莫大な数となるため、そのベクトル r_j を再現できる確率は極めて低い。このため、本発明を利用することによって、暗号の秘密性を高めることが可能となる。

【 0 0 9 4 】

【発明の効果】

本発明は、 $n - 1$ 次元回転行列から n 次元回転行列を生成するので、逐次処理にきわめて適合する。更に、本発明はその n 次元回転行列を用いて、 n 次元の空間の閉領域内に定義された n 次元ベクトルを回転並進させてカオス的かつ逐次的にベクトルを生成する非線形関数は実数で定義されているので、デジタル表現される任意表現データに対して暗号復号化を行うことができ、これにより、広範囲なアプリケーションに対して使用可能となる。

【図面の簡単な説明】

【図 1】

本発明の原理を説明するシステム構成図である。

【図 2】

図 1 の装置 1 0、1 2 が備える内部機能の例を説明する図である。

【図 3】

多次元ベクトルの生成処理を説明するフローチャートである。

【図 4】

多次元ベクトルの生成を利用した暗号復号方式を説明するフローチャートである。

【図 5】

本発明の実施態様の復号化処理を説明するフローチャートである。

【図 6】

本発明の実施態様における 3 次元ベクトル r_j の生成処理を説明するフローチャートである。

【図 7】

本発明の実施態様における、 n 次元回転行列 $R_n (\Omega_n)$ の生成処理を説明するフローチャートである。

【図 8】

本発明の実施態様における、3 次元ベクトルの回転操作の様子を説明する図である。

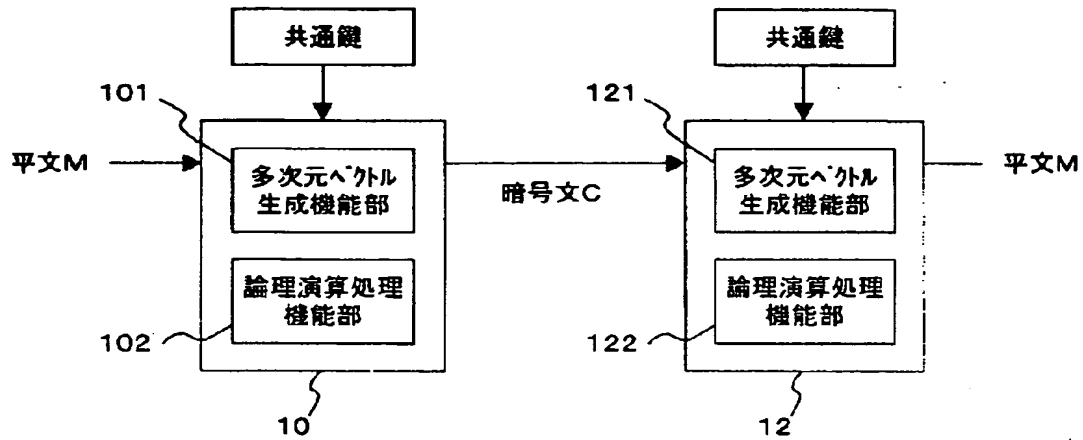
【符号の説明】

- 1 主プログラム
- 2 パラメータリスト生成ライブラリ
- 3 暗号化／復号化エンジン
- 4 多次元ベクトル回転関数発生ライブラリ
- 1 0 暗号化装置
- 1 2 復号化装置
- 1 0 1、1 2 1 多次元ベクトル生成機能部
- 1 0 2、1 2 2 倫理演算処理機能部

【書類名】 図面

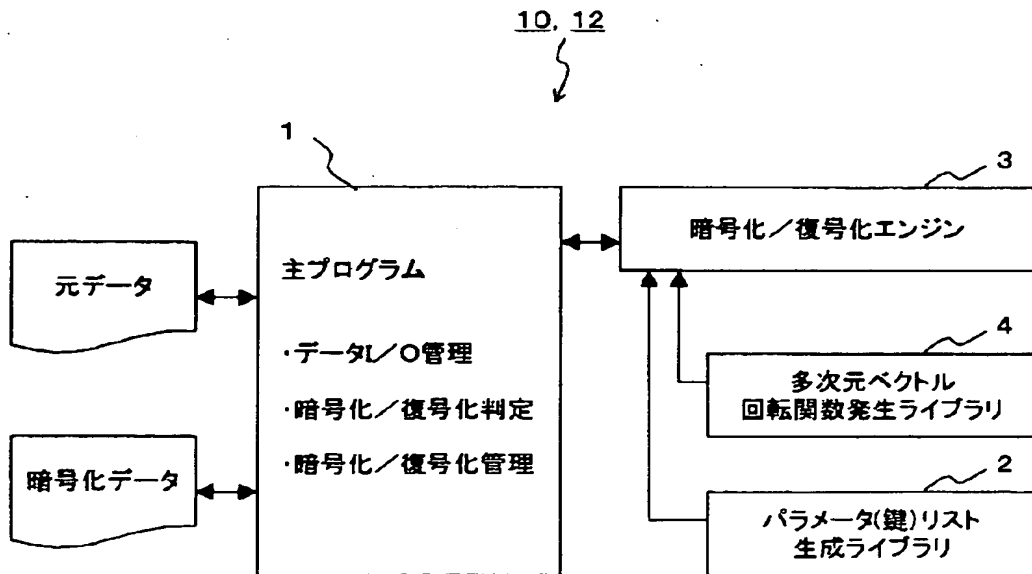
【図 1】

本発明の原理を説明するシステム構成図



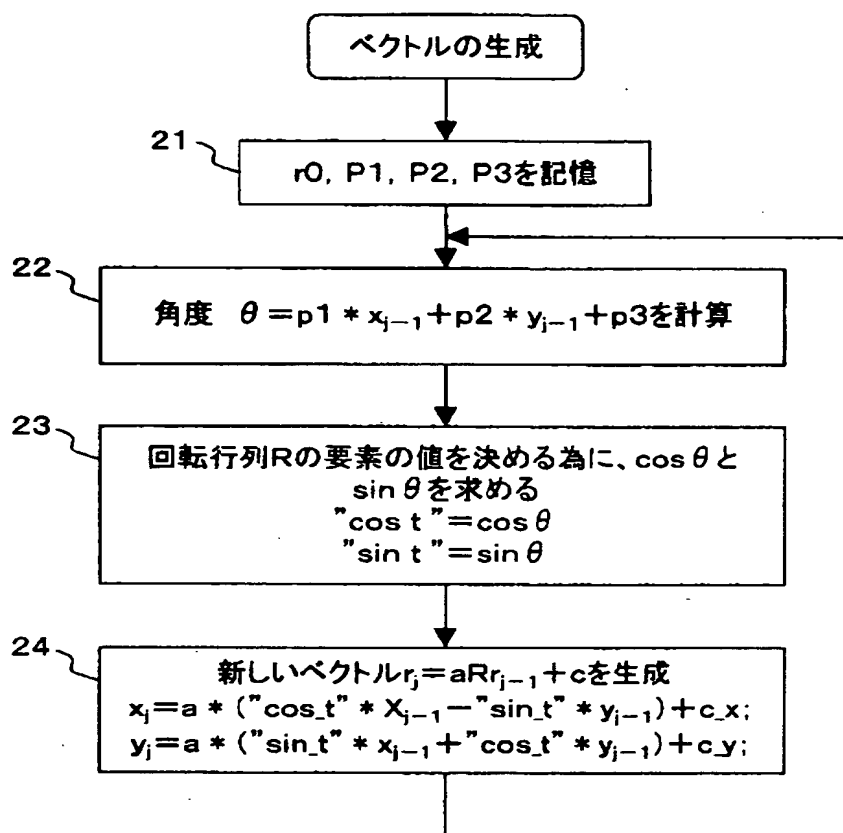
【図 2】

図1の装置10、12が備える内部機能の例を説明する図



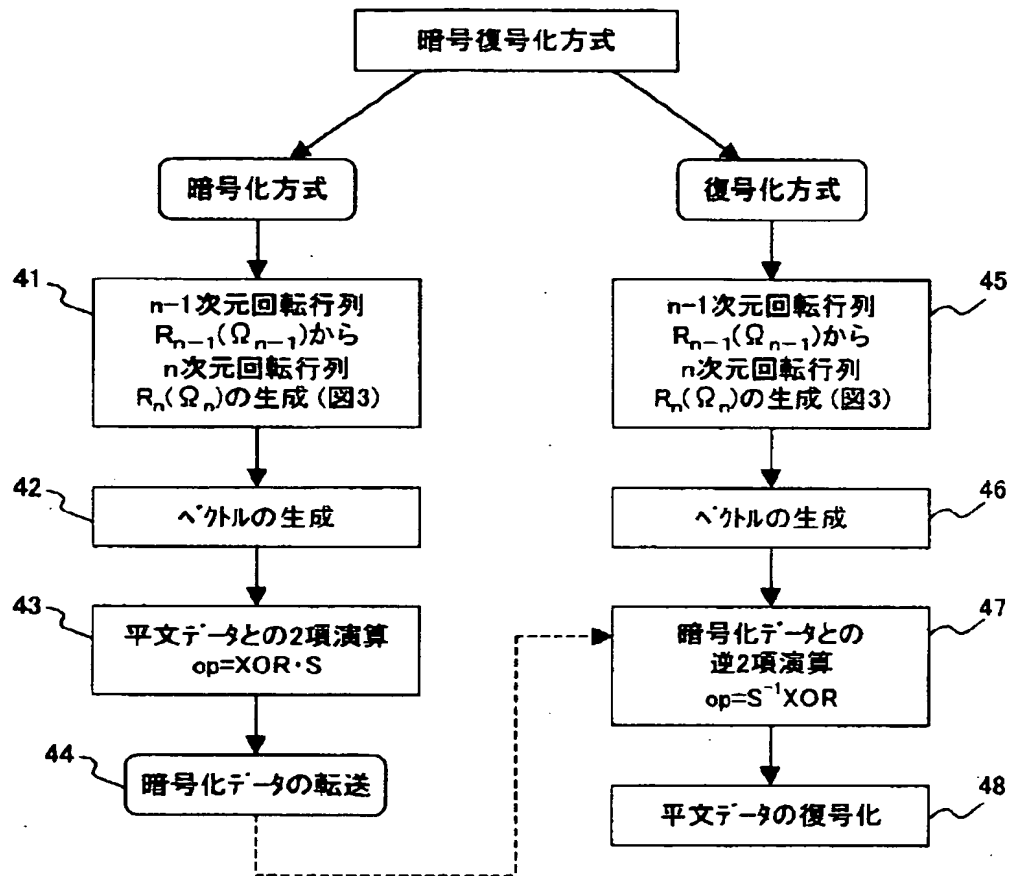
【図 3】

多次元ベクトルの生成処理を説明するフローチャート



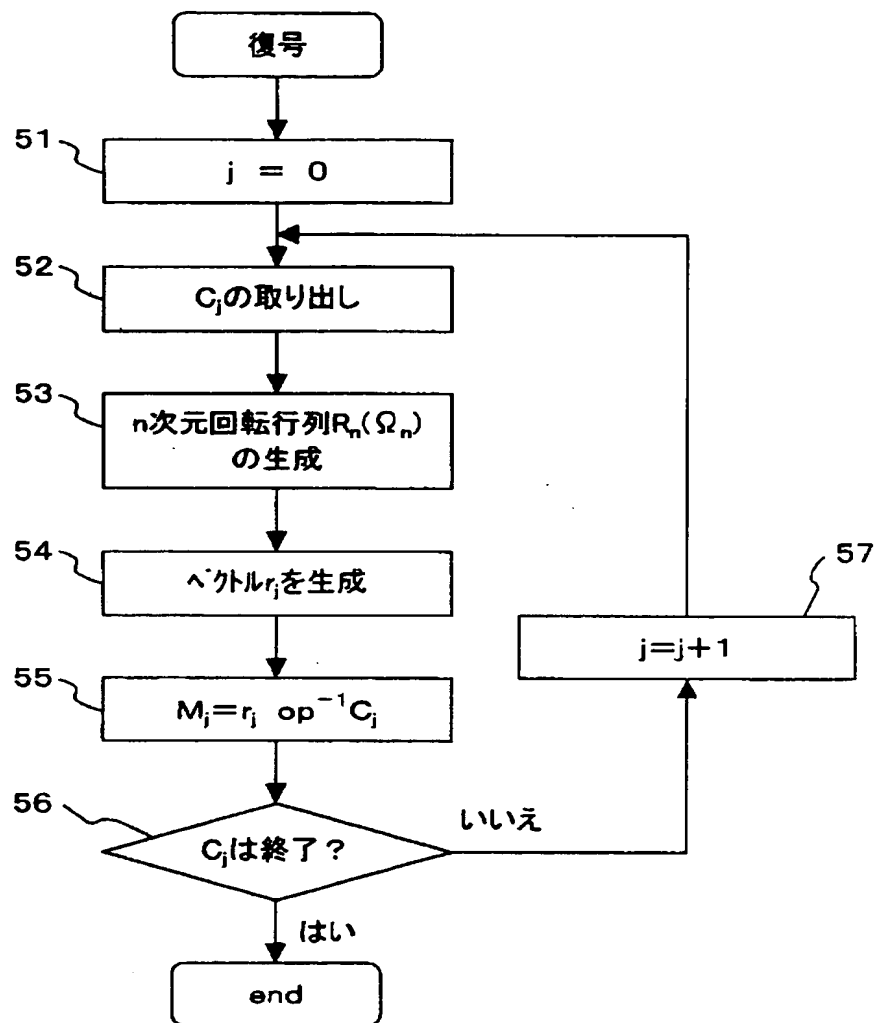
【図 4】

多次元ベクトルの生成を利用した暗号復号方式を説明するフローチャート

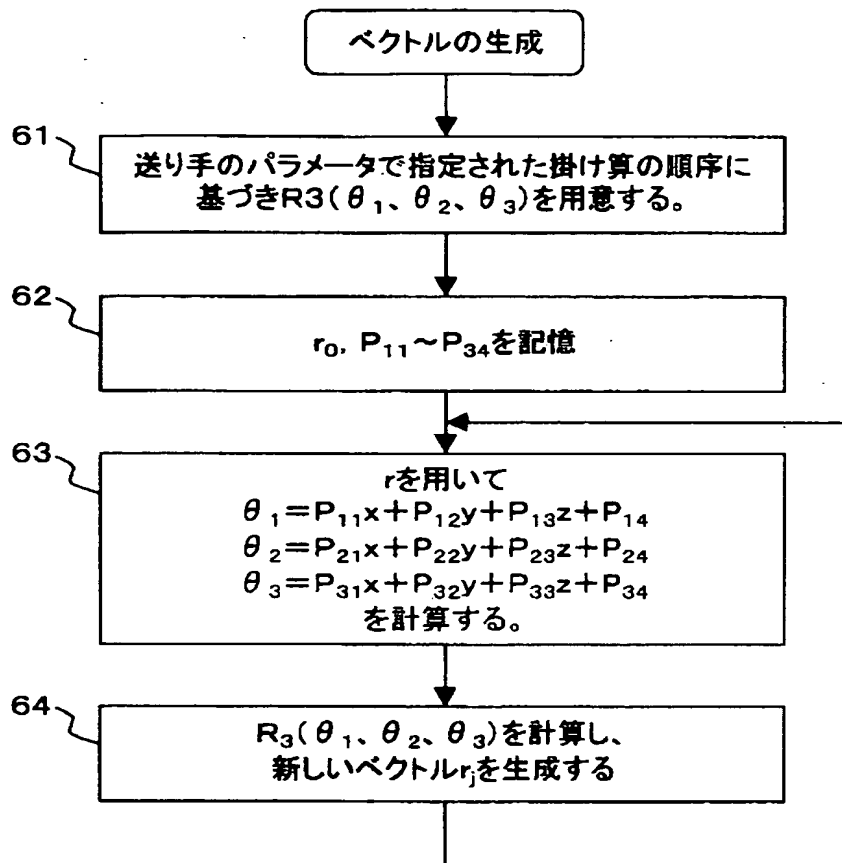


【図 5】

本発明の実施態様の復号化処理を説明するフローチャート

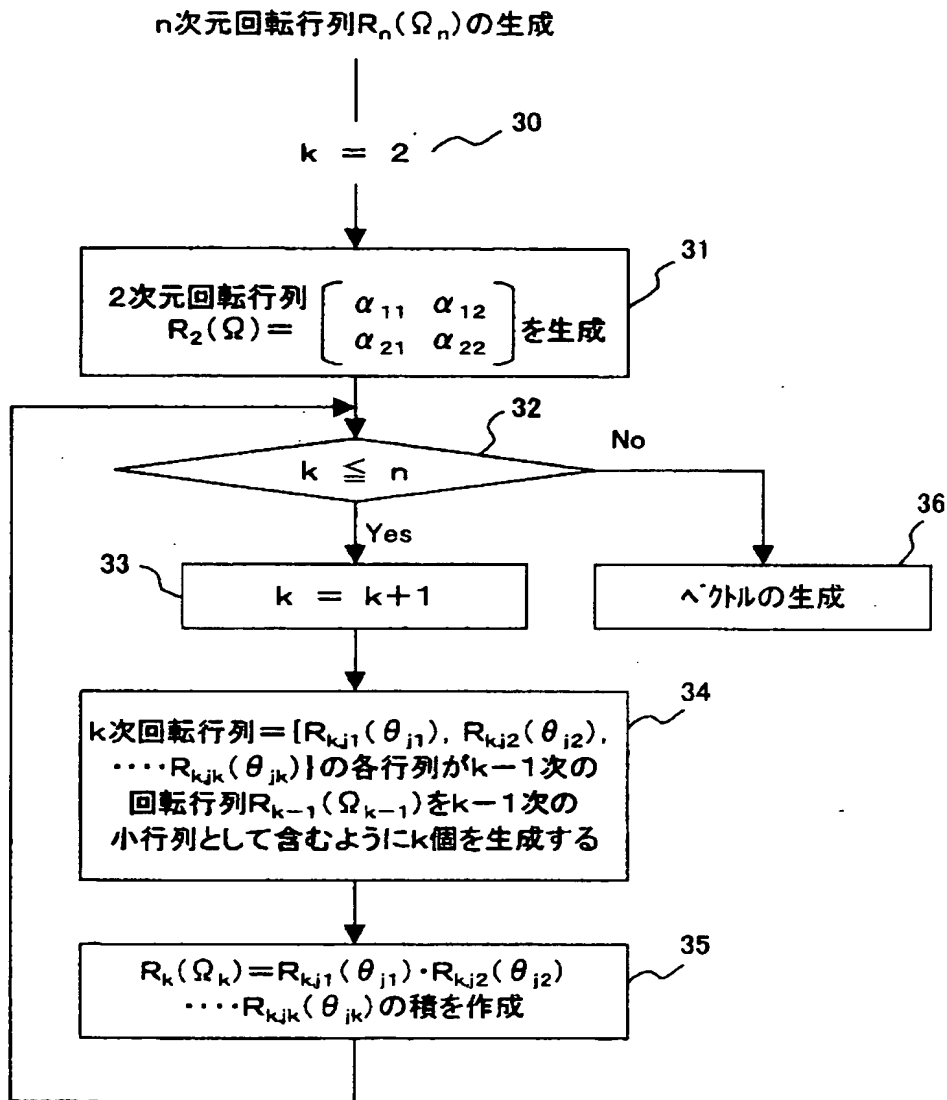


【図 6】

本発明の実施態様における3次元ベクトル r_j の
生成処理を説明するフローチャート

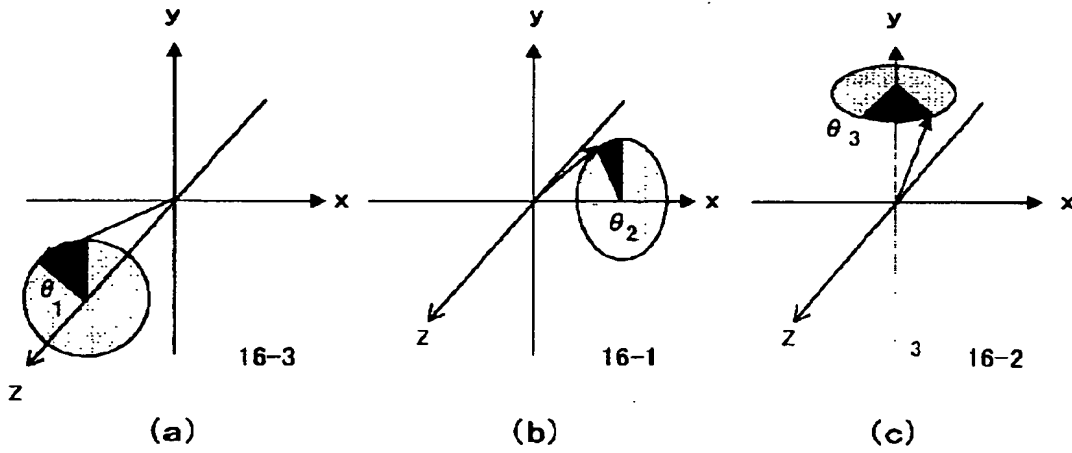
【図 7】

本発明の実施態様における、 n 次元回転行列 $R_n(\Omega_n)$ の
生成処理を説明するフローチャート



【図 8】

本発明の実施態様における、3次元ベクトルの
回転操作の様子を説明する図



【書類名】 要約書

【要約】

【課題】 信頼性が高く、かつ、広範囲なアプリケーションに適用可能な暗号化／復号化方法を実現する。

【解決手段】 装置 1 0 は、 n ($n \geq 1$) 次元の空間の閉領域内に定義された n 次元ベクトルを回転並進させる非線形関数を用いてカオス的かつ逐次的にベクトルを生成し、このベクトルを用いて平文 M を暗号化する。これにより生成された暗号文 C は装置 1 2 に送信される。装置 1 2 は、装置 1 0 と同様にして上記ベクトルを生成し、このベクトルを用いて暗号文 C を復号化する。装置 1 0、1 2 は、共通鍵を共有し、上記非線形関数は該共通鍵に対応するパラメータによって生成される。上記非線形関数は、前記 n 次元ベクトルを回転させる回転行列を含み、該ベクトルは互いに一致しないように逐次的に生成される。

【選択図】 図 1

出 願 人 履 歴 情 報

識別番号 [000001443]

1. 変更年月日 1998年 1月 9日
[変更理由] 住所変更
住 所 東京都渋谷区本町1丁目6番2号
氏 名 カシオ計算機株式会社